DL輪読会

Convolutional Sequence to Sequence Learning

2017/05/19
松尾研究室
M1 中川 大海

Agenda

1. Information

- Author
  - Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, Yann N. Dauphin
  - FAIR(Facebook AI Research)

- Submitted for arXiv on 8 May 2017

- Summary
  - fully convolutionalなseq2seqモデル
  - GLU, multi-hop attention, residual connectionなどの機構
  - GNMTより精度高くて9倍くらい早い
  - 実装がGitHubに上がってます(https://github.com/facebookresearch/fairseq)

2. Introduction

- 翻訳界隈でGNMT(Google Neural Machine Translation)が話題
  - encoder-decoder, bi-directional encoder, attention，LSTMブロック積んでresidual層で勾配消失防ぐ

- 一方、自然言語処理界隈では最近は並列計算できるCNNを用いるモデルが流行り
  - RNN: 並列計算できない、系列が長くなると勾配消失しやすい
  - CNN: 並列計算できるため計算高速化が可能、離れた系列間の関係も学習しやすい

- これまでもCNNを用いた手法は数々存在し、以下のような系譜をたどっている
  1. 精度は勝てないけど計算は早くなる
  2. 限られたデータセットでなら勝てる: [Bradbury et al. (2016), Kalchbrenner et al. (2016)]
  3. 多様なデータセットで勝てる: [Gehring et al. (2016), Dauphin et al. (2016)]
     - not fully convolutional
     - not generative model like seq2seq

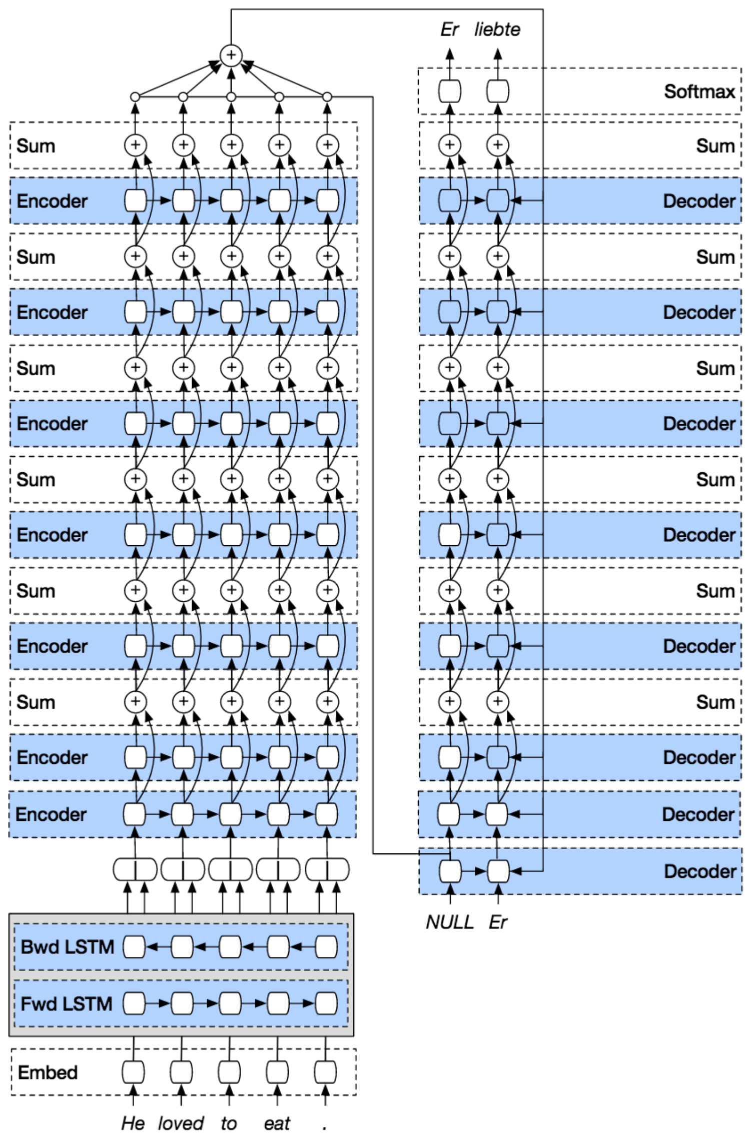
3. Related Work: 自然言語処理におけるタスク

- 識別
  - language modeling(言語モデル)
  - sentence classification
  - sentiment analysis
  - etc
- 生成
  - **sequence to sequence learning**
    - 翻訳, 要約
  - caption generation
  - etc

- 評価指標
  - Accuracy
  - PPL(Perplexity)
    - 単語の平均分岐数
    - $2^{(1\text{単語あたりのエントロピー})}$
    - どれぐらい単語を特定しにくいか(＝小さいほどよい)
  - BLEU(Bilingual Evaluation Understudy)
    - 正解(プロの翻訳)と予測の類似度的な指標
    - 大きいほどよい
  - …

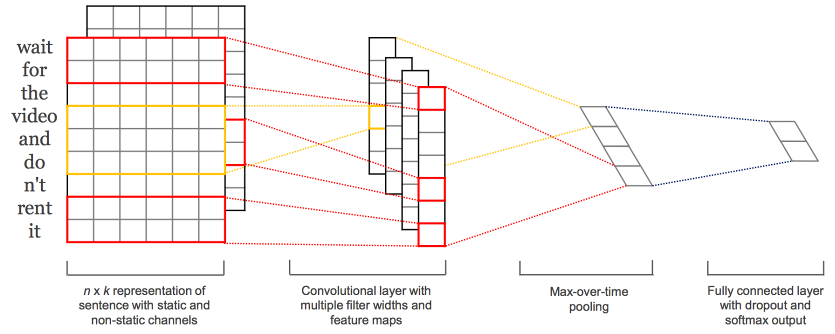
3. Related Work: GNMT [Wu et al. 2016]

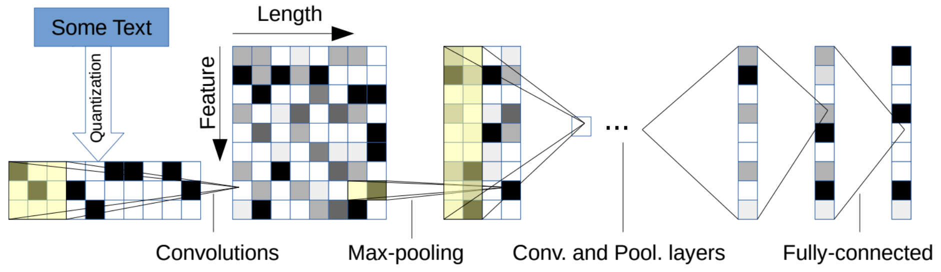
- encoder-decoderモデル
  - encode: 翻訳元の言語から潜在状態
  - decode: 潜在状態から翻訳先の言語へ
- 一層目のみbi-directional encoder
  - 初めの方の単語も文脈情報考慮できる
  - encoderの精度はattentionの効果にも影響
- attention
  - 入力系列のどこに注目して訳すればよいかまで学習
  - 計算時間は増えるが長い系列に特に有効
- 各層をresidualに
  - 出力H(x)でなく残差関数F(x)=H(x)-xを学習
  - 層を増やしても勾配消失しにくい
  - 入力をそのまま出力に加算するだけで実装できる

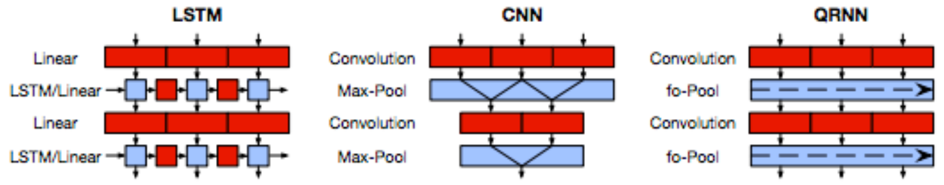


3. Related Work: CNNを活用した自然言語処理モデル

- Sentence Classification [Kim, 2014]
- Character-level Text classification [Zhang et al. 2015]
- Quasi-RNN [Bradbury et al. 2016]
  - LSTMライクにプーリング
- その他いろいろあります
  - http://kskskskss2.hatenadiary.jp/entry/20170122/1485082800
  - http://deeplearning.hatenablog.com/entry/neural_machine_translation_theory#seq2seq
  - https://www.slideshare.net/sheemap/convolutional-neural-netwoks
- 計算は高速化されるが、LSTMベースより精度が良かったり悪かったり、有効なデータセットが限られていたり

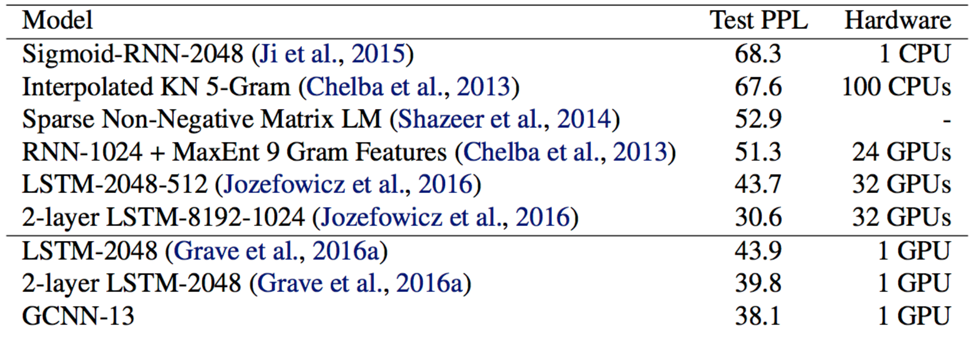

wait
for
the
video
and
do
n't
rent
it
n x k representation of sentence with static and non-static channels
Convolutional layer with multiple filter widths and feature maps
Max-over-time pooling
Fully connected layer with dropout and softmax output





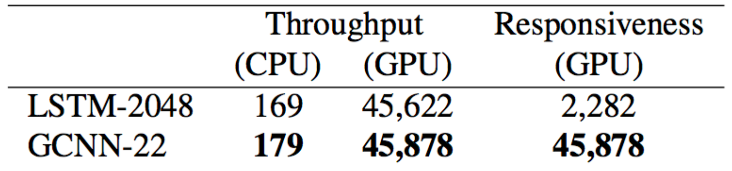
3. Related Work: CNNを活用した自然言語処理モデル

- Language Modeling with Gated CNN [Dauphin et al. 2016]
  - Gated Linear Unitsをゲート関数として導入
  - Residual処理
  - WikiText-103のタスクでSoTAのPPL
  - LSTMベースの20倍の速度

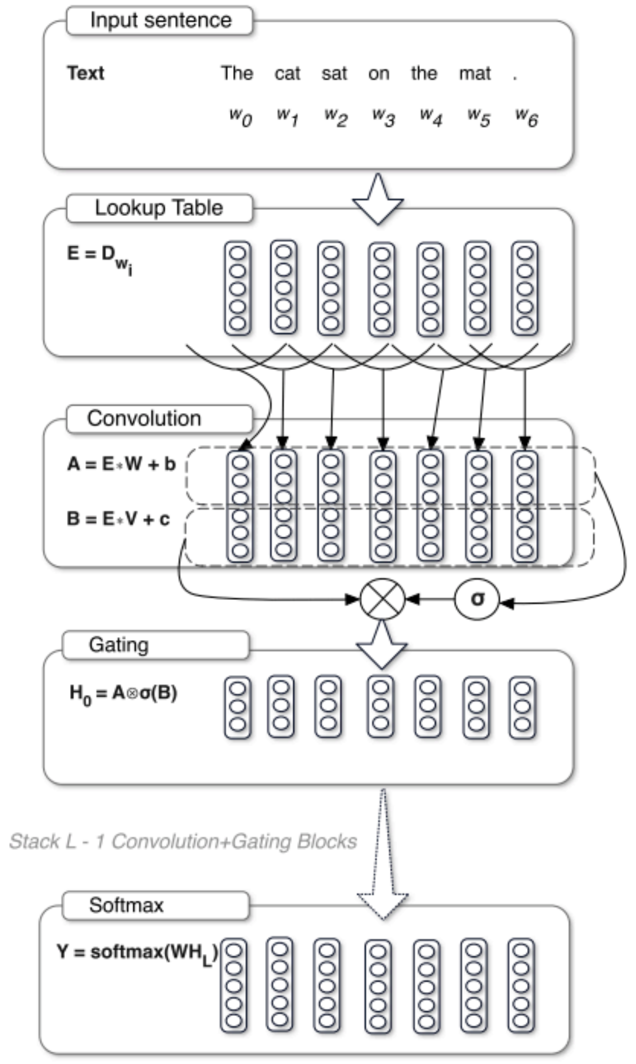


Model	Test PPL	Hardware
Sigmoid-RNN-2048 (Ji et al., 2015)	68.3	1 CPU
Interpolated KN 5-Gram (Chelba et al., 2013)	67.6	100 CPUs
Sparse Non-Negative Matrix LM (Shazeer et al., 2014)	52.9	-
RNN-1024 + MaxEnt 9 Gram Features (Chelba et al., 2013)	51.3	24 GPUs
LSTM-2048-512 (Jozefowicz et al., 2016)	43.7	32 GPUs
2-layer LSTM-8192-1024 (Jozefowicz et al., 2016)	30.6	32 GPUs
LSTM-2048 (Grave et al., 2016a)	43.9	1 GPU
2-layer LSTM-2048 (Grave et al., 2016a)	39.8	1 GPU
GCNN-13	38.1	1 GPU

Table 1. Results on the Google Billion Word test set.



	Throughput		Responsiveness
	(CPU)	(GPU)	(GPU)
LSTM-2048	169	45,622	2,282
GCNN-22	**179**	**45,878**	**45,878**



Input sentence
Text
The cat sat on the mat .
w_0 w_1 w_2 w_3 w_4 w_5 w_6
Lookup Table
$E = D_{w_i}$
Convolution
$A = E*W + b$
$B = E*V + c$
σ
Gating
$H_0 = A \otimes \sigma(B)$
Stack L - 1 Convolution+Gating Blocks
Softmax
$Y = \text{softmax}(WH_L)$

Figure 1. Architecture of the gated convolutional network for language modeling.

3. Related Work: CNNを活用したNMT

- Language Modeling with Gated CNN [Dauphin et al. 2016]
  - Gated Linear Unitsをゲート関数として導入
  - “allows the model to select which words or features are relevant to predict the next word.”
  - それまでの翻訳を踏まえて、その時点で文脈の特定の部分に着目するか広く見るか…などを表すゲート関数を学習できる
  - tanhベースのゲート関数よりも勾配が消失しにくい

$$h_l(\mathbf{X}) = (\mathbf{X} * \mathbf{W} + \mathbf{b}) \otimes \sigma(\mathbf{X} * \mathbf{V} + \mathbf{c})$$

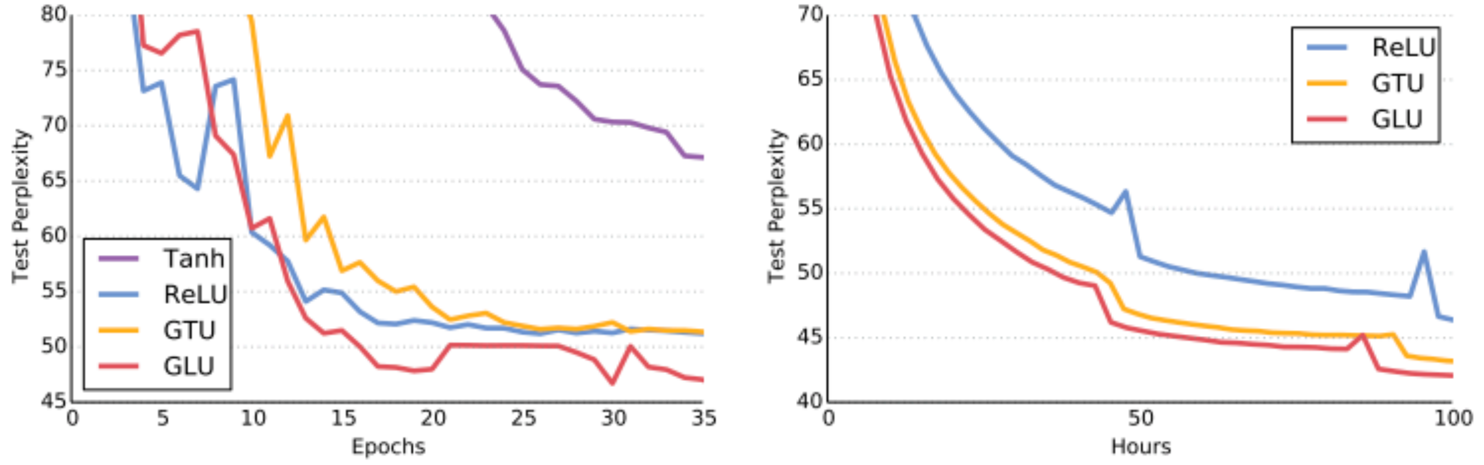


Figure 2. Learning curves on WikiText-103 (left) and Google Billion Word (right) for models with different activation mechanisms. Models with gated linear units (GLU) converge faster and to a lower perplexity.

4. Proposed Model

- やっていること
  1. 入力をembedding→畳み込みしてGLUに通す
     - decoder側も同様
  2. multi-hop attentionを計算
     - allow machines to reference different parts of text to build understanding during encoding.
  3. attentionつきの入力とdecoder contextsから予測

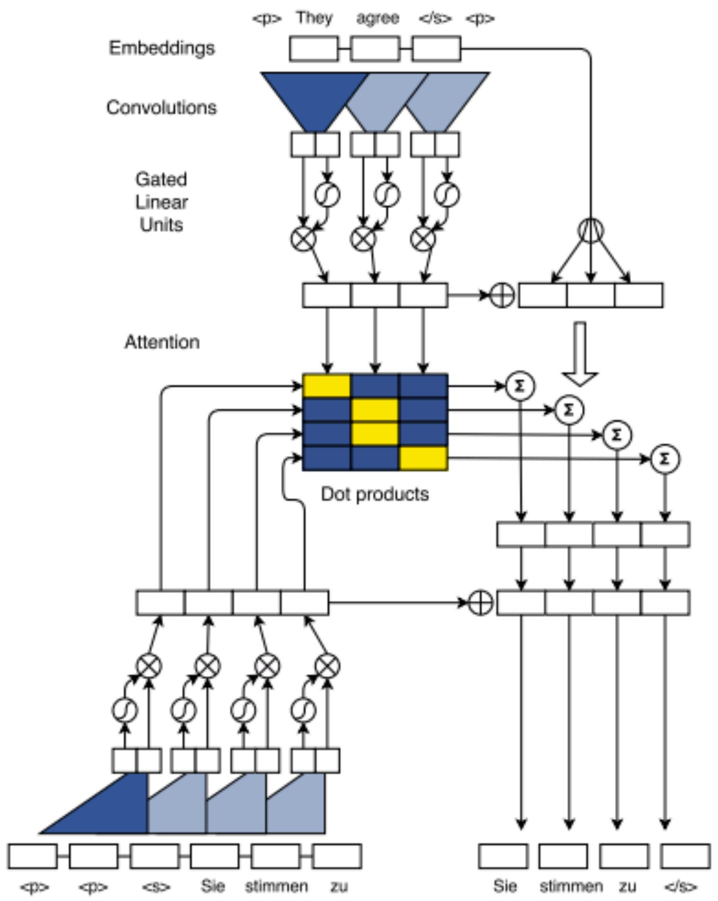


Figure 1. Illustration of batching during training. The English source sentence is encoded (top) and we compute all attention values for the four German target words (center) simultaneously. Our attentions are just dot products between decoder context representations (bottom left) and encoder representations. We add the conditional inputs computed by the attention (center right) to the decoder contexts which then predict the target words (bottom right). The sigmoid and multiplicative boxes illustrate Gated Linear Units.

4. Proposed Model

- やっていること
  1. 入力をembedding→畳み込みしてGLUに通す
     - decoder側も同様
  2. multi-hop attentionを計算
     - allow machines to reference different parts of text to build understanding during encoding.
  3. attentionつきの入力とdecoder contextsから予測

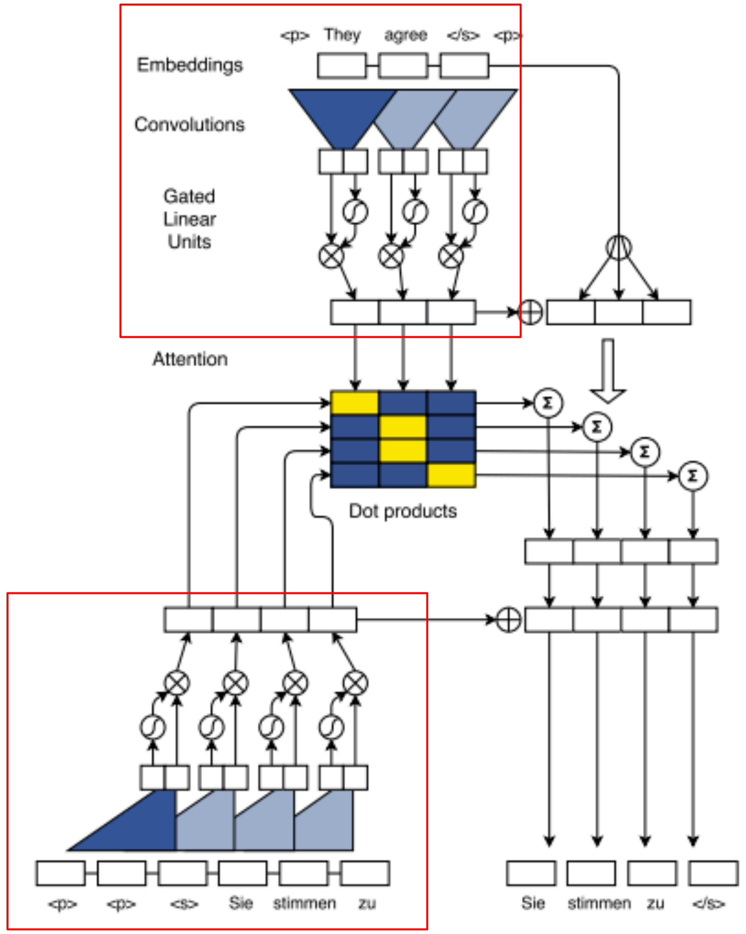


Figure 1. Illustration of batching during training. The English source sentence is encoded (top) and we compute all attention values for the four German target words (center) simultaneously. Our attentions are just dot products between decoder context representations (bottom left) and encoder representations. We add the conditional inputs computed by the attention (center right) to the decoder contexts which then predict the target words (bottom right). The sigmoid and multiplicative boxes illustrate Gated Linear Units.

4. Proposed Model

- やっていること
  1. 入力をembedding→畳み込みしてGLUに通す
     - decoder側も同様
  2. multi-hop attentionを計算
     - allow machines to reference different parts of text to build understanding during encoding.
  3. attentionつきの入力とdecoder contextsから予測

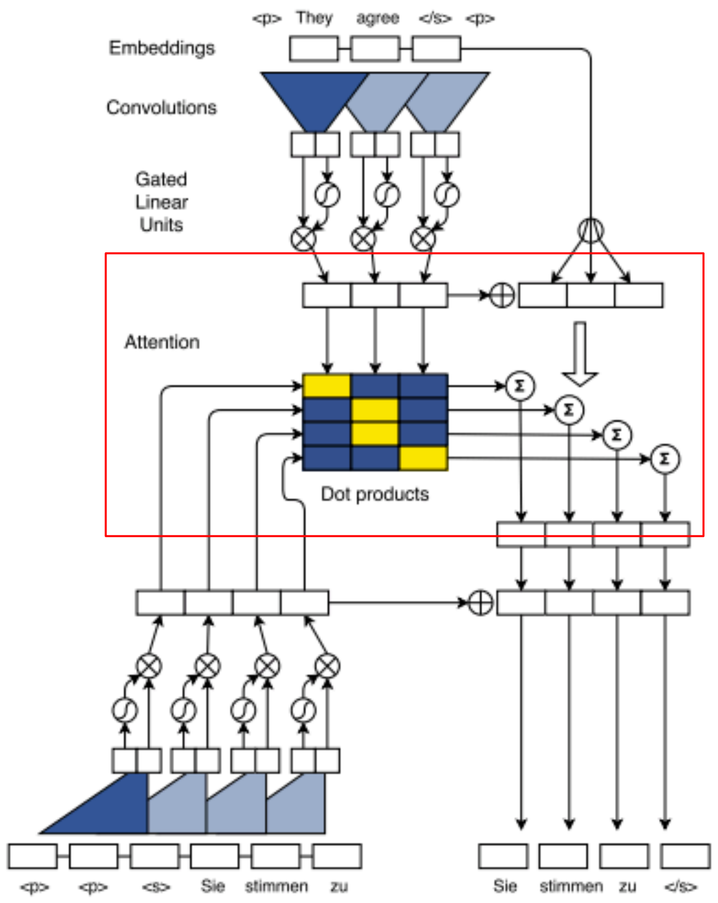


Figure 1. Illustration of batching during training. The English source sentence is encoded (top) and we compute all attention values for the four German target words (center) simultaneously. Our attentions are just dot products between decoder context representations (bottom left) and encoder representations. We add the conditional inputs computed by the attention (center right) to the decoder contexts which then predict the target words (bottom right). The sigmoid and multiplicative boxes illustrate Gated Linear Units.

4. Proposed Model

- やっていること

1. 入力をembedding→畳み込みしてGLUに通す
   - decoder側も同様
2. multi-hop attentionを計算
   - allow machines to reference different parts of text to build understanding during encoding.
3. attentionつきの入力とdecoder contextsから予測

→もう少し詳しく見ていきます

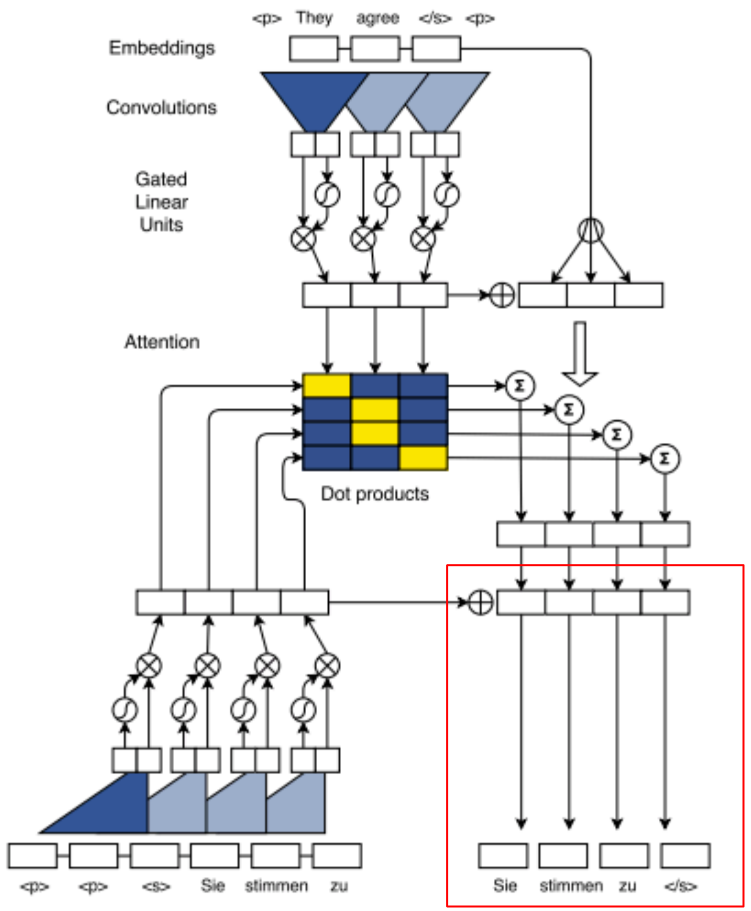


Figure 1. Illustration of batching during training. The English source sentence is encoded (top) and we compute all attention values for the four German target words (center) simultaneously. Our attentions are just dot products between decoder context representations (bottom left) and encoder representations. We add the conditional inputs computed by the attention (center right) to the decoder contexts which then predict the target words (bottom right). The sigmoid and multiplicative boxes illustrate Gated Linear Units.

4. Proposed Model

- やっていること
  1. 入力をembedding→畳み込みしてGLUに通す
     - decoder側も同様
- Position Embedding

input elements $\mathbf{x} = (x_1, \ldots, x_m)$ を embedding matrix $\mathcal{D} \in \mathbb{R}^{V \times f}$ によって

$\mathbf{w} = (w_1, \ldots, w_m)$, where $w_j \in \mathbb{R}^f$ にembedding。

position of input elements $\mathbf{p} = (p_1, \ldots, p_m)$ もconcatenateして

$\mathbf{e} = (w_1 + p_1, \ldots, w_m + p_m)$. とする。

– inputやoutputが文のどの部分を扱っているかの情報

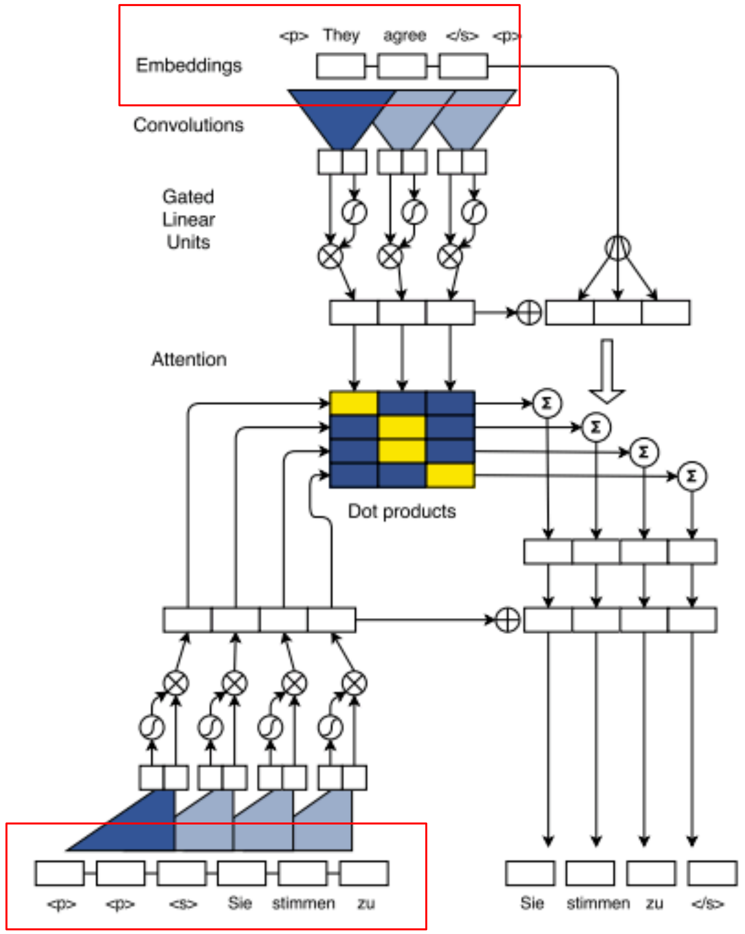


Figure 1. Illustration of batching during training. The English source sentence is encoded (top) and we compute all attention values for the four German target words (center) simultaneously. Our attentions are just dot products between decoder context representations (bottom left) and encoder representations. We add the conditional inputs computed by the attention (center right) to the decoder contexts which then predict the target words (bottom right). The sigmoid and multiplicative boxes illustrate Gated Linear Units.

4. Proposed Model

- やっていること
  1. 入力をembedding→畳み込みしてGLUに通す
     - decoder側も同様
- Convolution

  入力ベクトル $X \in \mathbb{R}^{k \times d}$ を

  $W \in \mathbb{R}^{2d \times kd}$, $b_w \in \mathbb{R}^{2d}$ のカーネルで畳み込んで $Y \in \mathbb{R}^{2d}$ とする。

  – 各隠れ層でresidual処理を行っている

$$h_i^l = v(W^l[h_{i-k/2}^{l-1}, \ldots, h_{i+k/2}^{l-1}] + b_w^l) + h_i^{l-1}$$

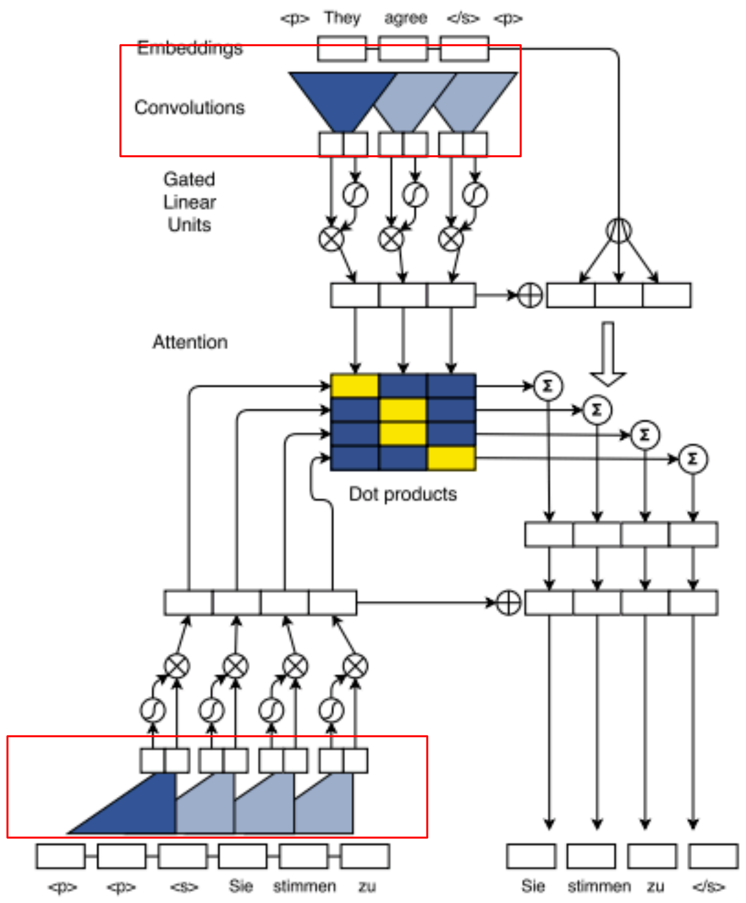


Figure 1. Illustration of batching during training. The English source sentence is encoded (top) and we compute all attention values for the four German target words (center) simultaneously. Our attentions are just dot products between decoder context representations (bottom left) and encoder representations. We add the conditional inputs computed by the attention (center right) to the decoder contexts which then predict the target words (bottom right). The sigmoid and multiplicative boxes illustrate Gated Linear Units.

4. Proposed Model

- やっていること
  1. 入力をembedding→畳み込みしてGLUに通す
     - decoder側も同様
- Gated Linear Units

  $Y = [A\ B] \in \mathbb{R}^{2d}$ から $v([A\ B]) = A \otimes \sigma(B)$ へ変換。

  – σ(B) controls which inputs A of the current context are relevant

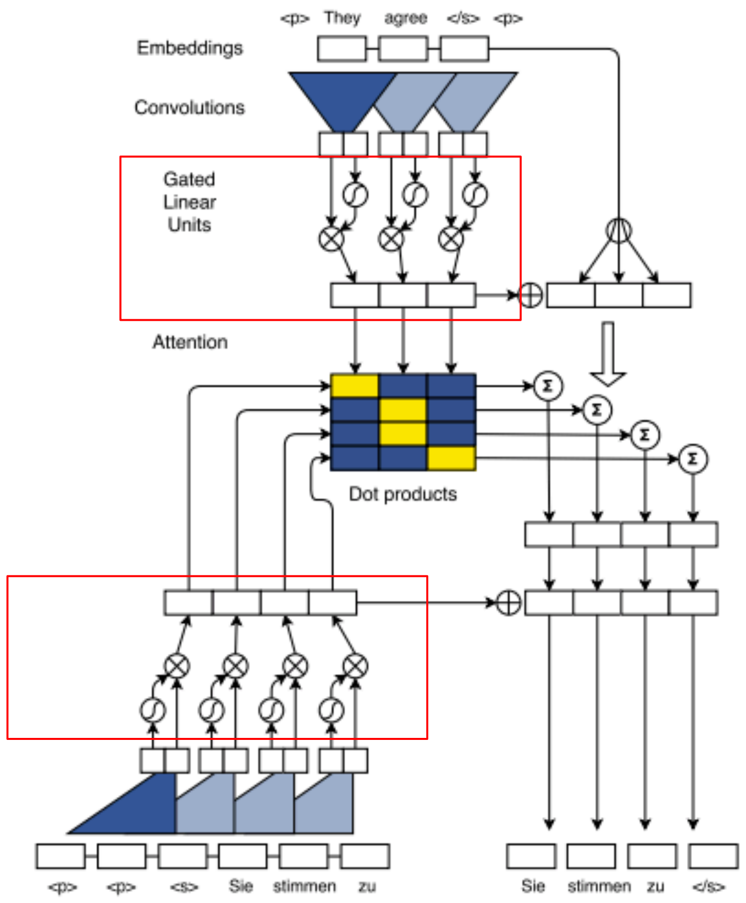


Figure 1. Illustration of batching during training. The English source sentence is encoded (top) and we compute all attention values for the four German target words (center) simultaneously. Our attentions are just dot products between decoder context representations (bottom left) and encoder representations. We add the conditional inputs computed by the attention (center right) to the decoder contexts which then predict the target words (bottom right). The sigmoid and multiplicative boxes illustrate Gated Linear Units.

4. Proposed Model

- やっていること

  2. multi-hop attentionを計算

  – current decoder state h_i^l とprevious target element g_i から attention score $d_i^l = W_d^l h_i^l + b_d^l + g_i$ を求める

  – decoder state summary d_i^l とoutput of the last encoder z_j^u からattention $a_{ij}^l = \frac{\exp\left(d_i^l \cdot z_j^u\right)}{\sum_{t=1}^m \exp\left(d_i^l \cdot z_t^u\right)}$ を求める

  – conditional input $c_i^l = \sum_{j=1}^m a_{ij}^l (z_j^u + e_j)$ を求める

z: large input context
e: point information
zがkey、z+eがvalueとして
key-value memory networkのように働くらしい

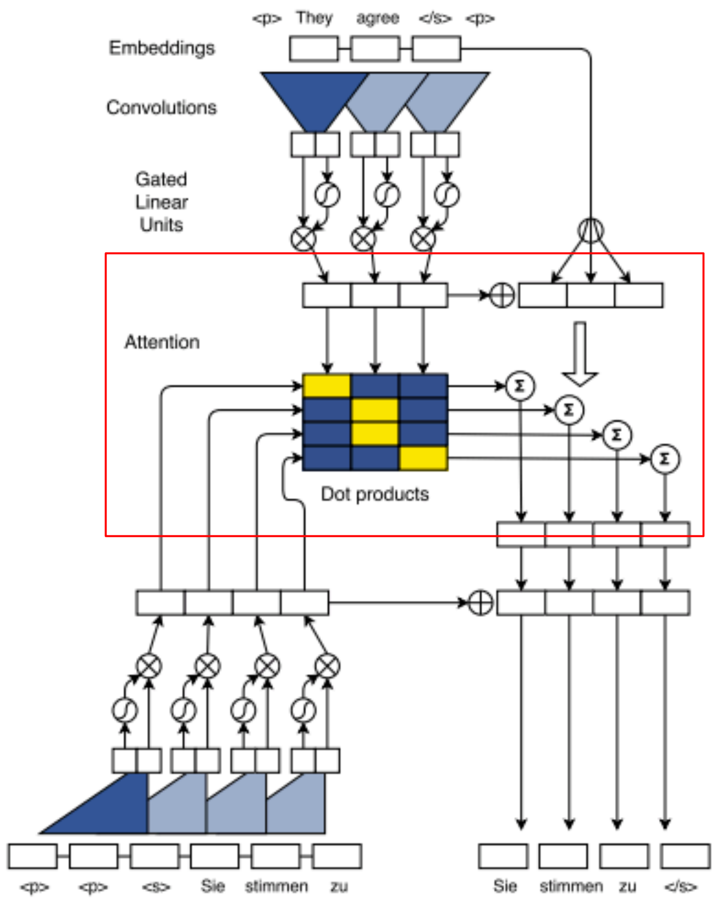


Figure 1. Illustration of batching during training. The English source sentence is encoded (top) and we compute all attention values for the four German target words (center) simultaneously. Our attentions are just dot products between decoder context representations (bottom left) and encoder representations. We add the conditional inputs computed by the attention (center right) to the decoder contexts which then predict the target words (bottom right). The sigmoid and multiplicative boxes illustrate Gated Linear Units.

4. Proposed Model

- やっていること
  2. multi-hop attentionを計算
  - decoder layer h_i^l はk-1個のattention historyにアクセスできる
    - $h_{i-k}^{l-1}, \ldots, h_i^{l-1}$ に $c_{i-k}^{l-1}, \ldots, c_i^{l-1}$ が含まれるため
  - 過去のattention情報を反映しやすい
    - RNNだと消失しやすい
  - https://code.facebook.com/posts/1978007565818999/a-novel-approach-to-neural-machine-translation/

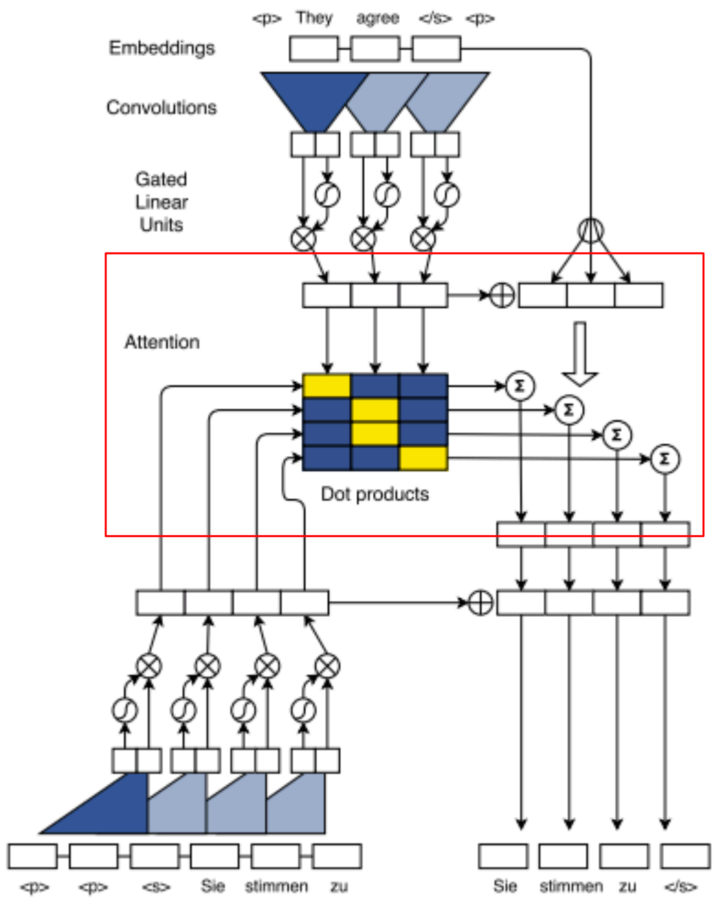


Figure 1. Illustration of batching during training. The English source sentence is encoded (top) and we compute all attention values for the four German target words (center) simultaneously. Our attentions are just dot products between decoder context representations (bottom left) and encoder representations. We add the conditional inputs computed by the attention (center right) to the decoder contexts which then predict the target words (bottom right). The sigmoid and multiplicative boxes illustrate Gated Linear Units.

4. Proposed Model

- やっていること
  3. attentionつきの入力とdecoder contextsから予測

$$p(y_{i+1}|y_1,\dots,y_i,\mathbf{x}) = \mathrm{softmax}(W_o h_i^L + b_o) \in \mathbb{R}^T$$

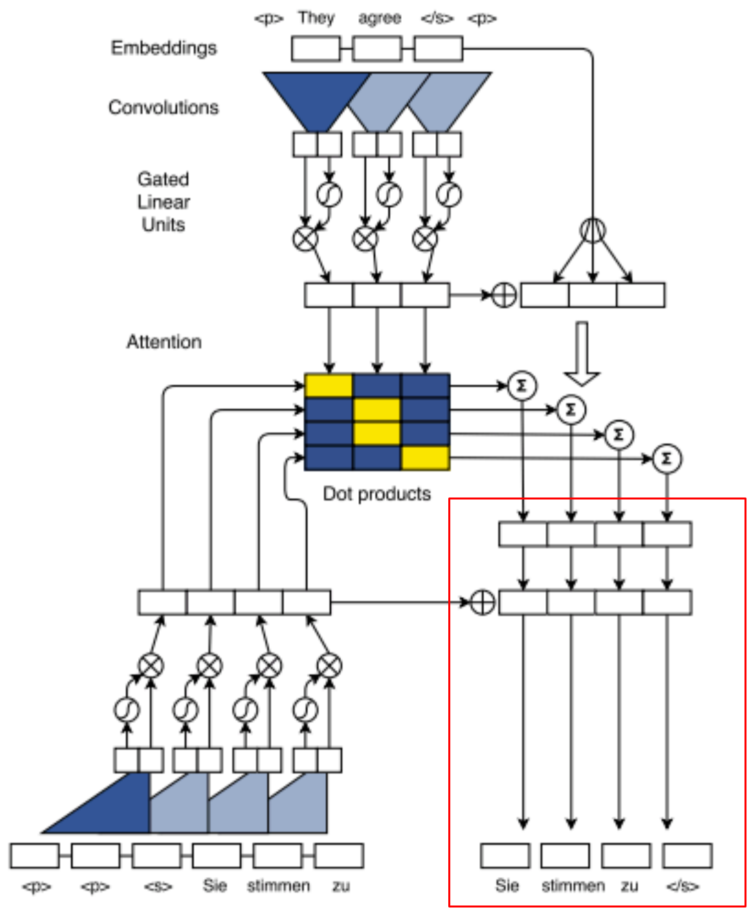


Figure 1. Illustration of batching during training. The English source sentence is encoded (top) and we compute all attention values for the four German target words (center) simultaneously. Our attentions are just dot products between decoder context representations (bottom left) and encoder representations. We add the conditional inputs computed by the attention (center right) to the decoder contexts which then predict the target words (bottom right). The sigmoid and multiplicative boxes illustrate Gated Linear Units.

5. Experiments & Results

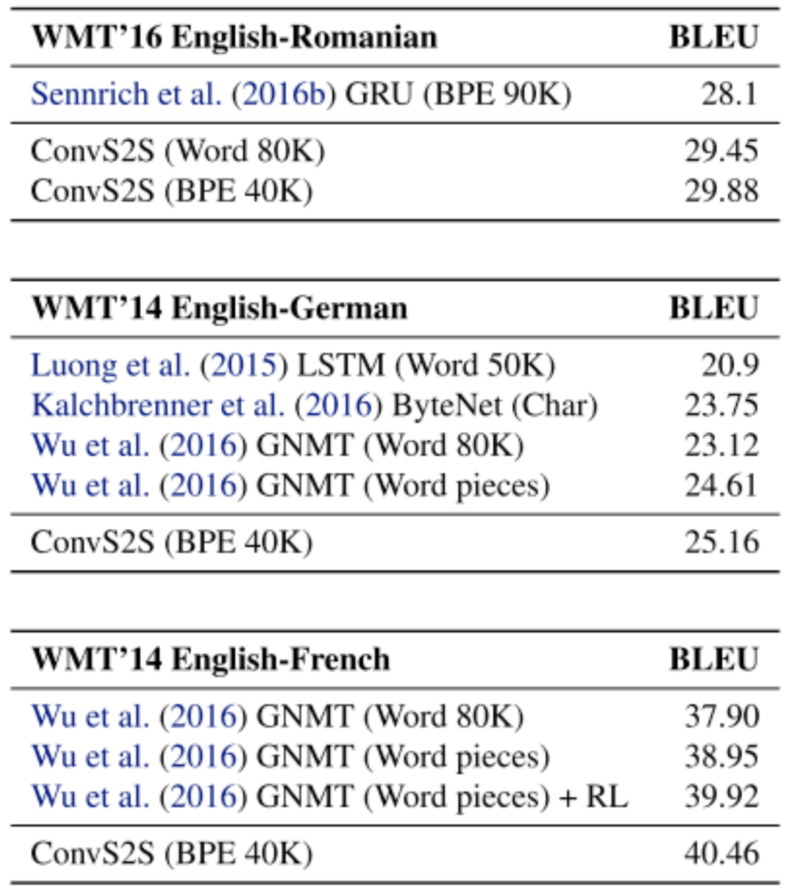
- Translation task
  - Datasets
    - WMT’16 English-Romanian, WMT’14 English-German, WMT’14 English-French
  - Experiment 1: Recurrent vs. Convolutional
    - LSTMベースのモデルたち, ByteNet, GNMT
  - Experiment2: Generation speed vs. GNMT
  - Experiment3: Effect of some architectures

5. Experiments & Results

- Summarization task
  - Datasets
    - Abstractive summarization (Gigaword corpus)
  - Compare Accuracy with RNN SoTA Models
    - Shen et al., 2016
    - Suzuki & Nagata, 2017

5. Results: Translation task

- 1. Recurrent vs. Convolutional
  - いずれのデータセットでも最良のBLEU



WMT'16 English-Romanian	BLEU
Sennrich et al. (2016b) GRU (BPE 90K)	28.1
ConvS2S (Word 80K)	29.45
ConvS2S (BPE 40K)	29.88

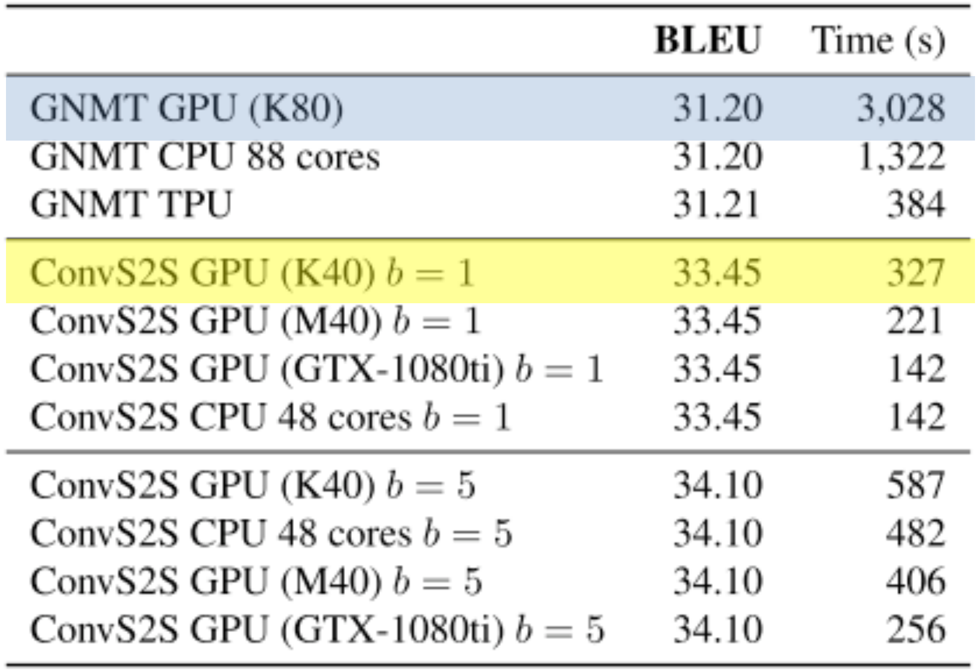
WMT'14 English-German	BLEU
Luong et al. (2015) LSTM (Word 50K)	20.9
Kalchbrenner et al. (2016) ByteNet (Char)	23.75
Wu et al. (2016) GNMT (Word 80K)	23.12
Wu et al. (2016) GNMT (Word pieces)	24.61
ConvS2S (BPE 40K)	25.16

WMT'14 English-French	BLEU
Wu et al. (2016) GNMT (Word 80K)	37.90
Wu et al. (2016) GNMT (Word pieces)	38.95
Wu et al. (2016) GNMT (Word pieces) + RL	39.92
ConvS2S (BPE 40K)	40.46

Table 1. Accuracy on WMT tasks comapred to previous work. Our results are averaged over three runs.

5. Results: Translation task

- 2. Generation speed vs. GNMT
  - 提案モデルのGPU(K40)でGNMTのGPU(K80)より高精度で9.3倍の速さ
    - K80はK40二つ分みたいなもの
    - “We did not have such a GPU available”
  - ビームサーチ幅(b)を広げるとスピードは多少落ちるが、BLEUは上がる
  - CPUはコア数が違うので比較できないとのこと

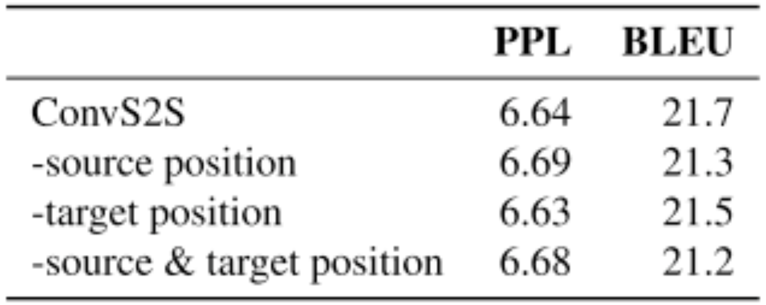


	BLEU	Time (s)
GNMT GPU (K80)	31.20	3,028
GNMT CPU 88 cores	31.20	1,322
GNMT TPU	31.21	384
ConvS2S GPU (K40) $b = 1$	33.45	327
ConvS2S GPU (M40) $b = 1$	33.45	221
ConvS2S GPU (GTX-1080ti) $b = 1$	33.45	142
ConvS2S CPU 48 cores $b = 1$	33.45	142
ConvS2S GPU (K40) $b = 5$	34.10	587
ConvS2S CPU 48 cores $b = 5$	34.10	482
ConvS2S GPU (M40) $b = 5$	34.10	406
ConvS2S GPU (GTX-1080ti) $b = 5$	34.10	256

Table 2. CPU and GPU generation speed in seconds on the development set of WMT’14 English-French. We show results for different beam sizes b. GNMT figures are taken from Wu et al. (2016). CPU speeds are not directly comparable because Wu et al. (2016) use a 88 core machine compared to our 48 core setup.

5. Results: Translation task

- 3. Effect of position embedding
  - position embeddingはあまり影響なし

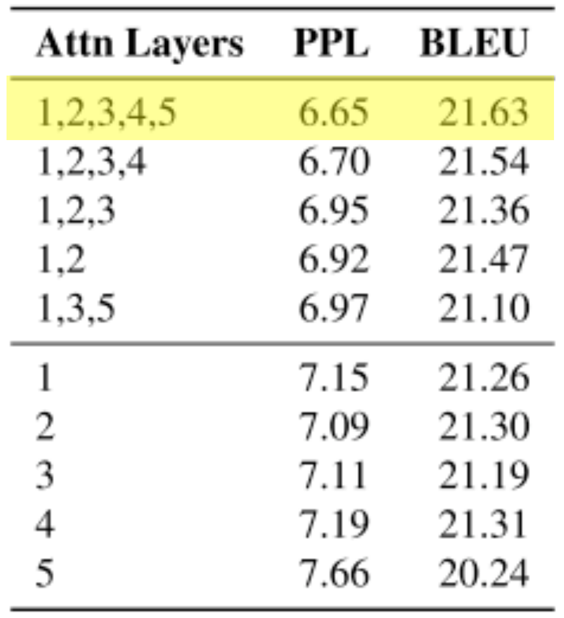


	PPL	BLEU
ConvS2S	6.64	21.7
-source position	6.69	21.3
-target position	6.63	21.5
-source & target position	6.68	21.2

Table 3. Effect of removing position embeddings from our model in terms of validation perplexity (valid PPL) and BLEU.

5. Results: Translation task

- 3. Effect of multi-step attention
  - decoder layer全てにattentionするのが最良
  - 計算的なoverheadもほとんどない

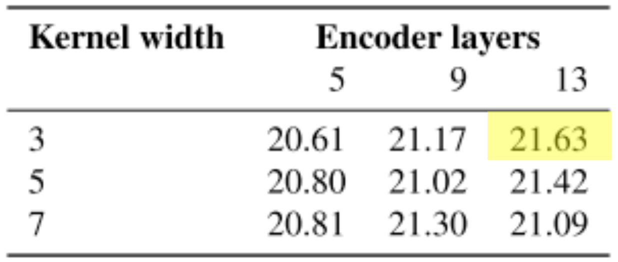


Attn Layers	PPL	BLEU
1,2,3,4,5	6.65	21.63
1,2,3,4	6.70	21.54
1,2,3	6.95	21.36
1,2	6.92	21.47
1,3,5	6.97	21.10
1	7.15	21.26
2	7.09	21.30
3	7.11	21.19
4	7.19	21.31
5	7.66	20.24

Table 4. Multi-step attention in all five decoder layers or fewer layers in terms of validation perplexity (PPL) and test BLEU.

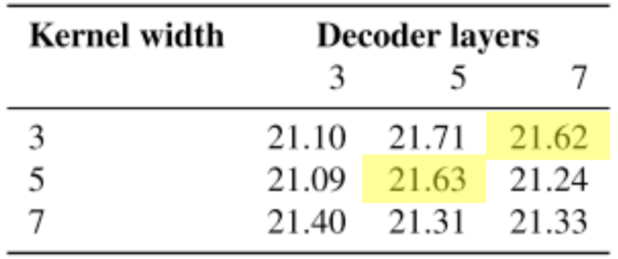
5. Results: Translation task

- 3. Effect of kernel size & depth
  - 狭く、深くが良い
  - Encoderは結構深くできる
  - Decoderはあまり効果なし



Kernel width	Encoder layers 5	9	13
3	20.61	21.17	21.63
5	20.80	21.02	21.42
7	20.81	21.30	21.09

Table 6. Encoder with different kernel width in terms of BLEU.



Kernel width	Decoder layers 3	5	7
3	21.10	21.71	21.62
5	21.09	21.63	21.24
7	21.40	21.31	21.33

Table 7. Decoder with different kernel width in terms of BLEU.

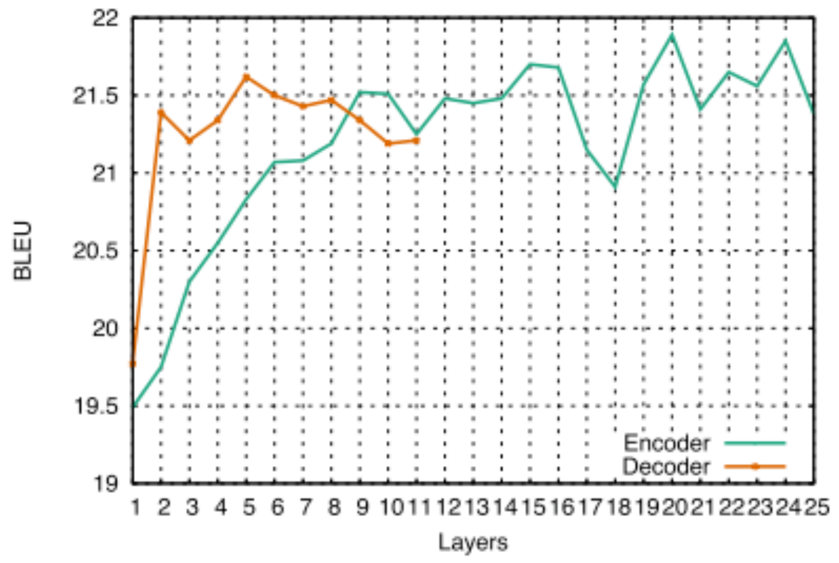
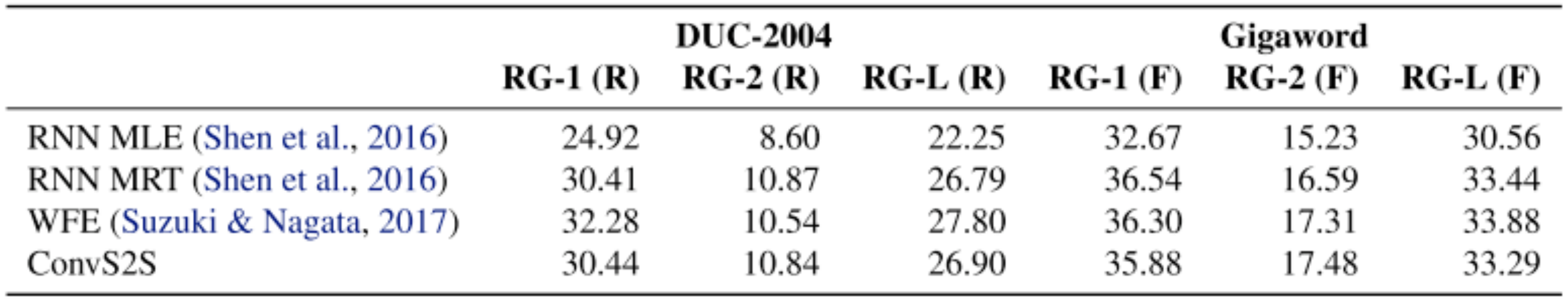


Figure 2. Encoder and decoder with different number of layers.

5. Results: Summarization task

- Accuracy
  - 勝ってるorそんなに負けてない(らしい)
  - 比較対象のモデルはいろいろspecificな加工してる
    - けど、手を加えていない提案モデルでも同じくらいの精度を出せている、とのこと
  - 提案モデルにも同様の処理はできる(らしい)



		DUC-2004			Gigaword	
	RG-1 (R)	**RG-2 (R)**	**RG-L (R)**	**RG-1 (F)**	**RG-2 (F)**	**RG-L (F)**
RNN MLE (Shen et al., 2016)	24.92	8.60	22.25	32.67	15.23	30.56
RNN MRT (Shen et al., 2016)	30.41	10.87	26.79	36.54	16.59	33.44
WFE (Suzuki & Nagata, 2017)	32.28	10.54	27.80	36.30	17.31	33.88
ConvS2S	30.44	10.84	26.90	35.88	17.48	33.29

Table 5. Accuracy on two summarization tasks in terms of Rouge-1 (RG-1), Rouge-2 (RG-2), and Rouge-L (RG-L).

6. Conclusion

- fully convolutionalなseq2seqモデルを提案
  - GLU, residual connection, multi-hop attention(, position embedding)などの機構を活用

- seq2seqモデルでSoTAな精度&GNMTの9倍の速度を達成した

感想

- CNNすごい

- NMTとかattentionとか全然わかってなかったので勉強になりました
  - 参考資料見ると結構分かるようになると思います

参考文献

- Gehring, Jonas, et al. "Convolutional Sequence to Sequence Learning." *arXiv preprint arXiv:1705.03122* (2017).
- Kim, Yoon. "Convolutional neural networks for sentence classification." *arXiv preprint arXiv:1408.5882* (2014).
- Zhang, Xiang, Junbo Zhao, and Yann LeCun. "Character-level convolutional networks for text classification." *Advances in neural information processing systems*. 2015.
- Bradbury, James, et al. "Quasi-Recurrent Neural Networks." *arXiv preprint arXiv:1611.01576* (2016).
- Wu, Yonghui, et al. "Google's Neural Machine Translation System: Bridging the Gap between Human and Machine Translation." *arXiv preprint arXiv:1609.08144* (2016).
- Dauphin, Yann N., et al. "Language Modeling with Gated Convolutional Networks." *arXiv preprint arXiv:1612.08083* (2016).
- Shen, Shiqi, et al. "Neural Headline Generation with Sentence-wise Optimization." *arXiv preprint arXiv:1604.01904* (2016).
- Suzuki, Jun, and Masaaki Nagata. "Cutting-off Redundant Repeating Generations for Neural Abstractive Summarization." *EACL 2017* (2017): 291.

参考文献

- Facebook AI Researchによる説明
  - https://code.facebook.com/posts/1978007565818999/a-novel-approach-to-neural-machine-translation/
- NMT一般の参考
  - http://deeplearning.hatenablog.com/entry/neural_machine_translation_theory#seq2seq
- GNMTの解説
  - http://smerity.com/articles/2016/google_nmt_arch.html
  - http://www.yasuhisay.info/entry/2016/11/23/000000
- Residual層の解説
  - http://terada-h.hatenablog.com/entry/2016/12/13/192940
- Attentionの解説
  - https://www.slideshare.net/yutakikuchi927/deep-learning-nlp-attention
  - インタラクティブに理解できる→　http://distill.pub/2016/augmented-rnns/